Computers & Operations Research 39 (2012) 1577–1581

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

# Improving a Lagrangian decomposition for the unconstrained binary quadratic programming problem

Geraldo Regis Mauri [a,*], Luiz Antonio Nogueira Lorena [b]

[a] Federal University of Espírito Santo – UFES, Alto Universitário s/n, Guararema, Alegre-ES 29500-000, Brazil
[b] National Institute for Space Research – INPE, Av. dos Astronautas 1758, Jd. da Granja, São José dos Campos-SP 12210-970, Brazil

ARTICLE INFO

Available online 12 September 2011

*Keywords:*
Column generation
Lagrangian decomposition
Quadratic programming

ABSTRACT

This paper presents a new alternative of Lagrangian decomposition based on column generation technique to solve the unconstrained binary quadratic programming problem. We use a mixed binary linear version of the original quadratic problem with constraints represented by a graph. This graph is partitioned into clusters of vertices forming subproblems whose solutions use the dual variables obtained by a coordinator problem. Computational experiments consider a set of difficult instances and the results are compared against other methods reported recently in the literature.



## 1. Introduction

The unconstrained binary quadratic programming (*QP*) problem consists of maximizing (or minimizing) a quadratic objective function by the choice of suitable values for the binary decision variables. The *QP* is a classical *NP-hard* non-linear problem [1] and applications have been reported in several areas. *QP* still presents several problems modeled by graphs including maximum independent set, max-cut and others.

Methods based on tree-search algorithms to solve *QP* are found in the literature. Pardalos and Rodgers [2] report a tree-search method that uses bounds based on the variables fixing at each node of the tree presenting results for problems with up to 200 variables. Billionnet and Sutter [3] present a tree-search method to solve problems with up to 100 variables. Huang et al. [4] describe different strategies for estimating lower bounds for a minimization quadratic binary programming problem. These strategies are based on a branch and bound and the results are reported for problems with up to 60 variables. We can also find some exact methods publicly available, such as the Biq Mac method [5].

Glover et al. [6] present several heuristics to solve *QP*. These heuristics are tested using instances with up to 2500 variables. Palubeckis [7] presents five different multistart tabu search strategies to solve *QP*. Results are reported for instances with up to 6000 variables, and the computational times showed to be smaller than other methods found in the literature. Billionet and Elloumi [1] apply some convexification techniques on the objective function and solve it by CPLEX. Results are reported by instances with up to 200 variables.

The *QP* linearization is a common practice to get an equivalent linear model whose solutions are feasible to the original quadratic model [8]. *QP* is converted into a mixed integer linear problem and the linear relaxation of its decision variables allows finding an upper bound for the original problem. This bound is known as *roof dual* [9], and this approach can define bounds for the optimal solution allowing to evaluate the best feasible solution found over the optimal.

Adams and Dearing [9] discuss alternatives about getting bounds for *QP*. Other strategies for linearization of *QP* are presented and discussed in [10,8]. Chardaire and Sutter [11] propose an algorithm based on a Lagrangian decomposition of the original quadratic objective function in a sum of pseudo-linear functions. Their results show good relaxation bounds, but the gaps increase significantly with the problem size. They show that the Lagrangian decomposition results are better than other decomposition methods. Besides, they show that the proposed algorithm gets the *roof dual* in the worst case. The results are presented for problems with up to 100 variables.

Alternative Lagrangian decomposition approaches are reported in [12]. These approaches improve the one presented in [11] reporting results for problems with up to 500 variables. Mauri and Lorena [12] also report other relaxation methods to find upper bounds for *QP*: a linear relaxation of *QP* (*roof dual*), a linear relaxation using cut constraints and alternative traditional Lagrangian relaxations.

This paper proposes a new alternative based on column generation (*CG*) to solve a Lagrangian decomposition approach reported recently in [12] to find upper bounds and feasible approximate solutions for *QP*. We propose to replace the subgradient algorithm by a *CG* on solving the dual problem for the *QP* decomposition.

* Corresponding author. Tel.: +55 28 3552 8921; fax: +55 28 3552 8903.
*E-mail addresses:* mauri@cca.ufes.br, grmauri@gmail.com (G.R. Mauri), lorena@lac.inpe.br (L.A.N. Lorena).

0305-0548/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2011.09.008

So we have a coordinator problem formed by relaxed constraints and columns are generated by subproblems using dual variables.

The remaining of the paper is organized as follows. Section 2 reports in detail the Lagrangian decomposition presented in [12]. The proposed column generation is described in Section 3. Computational results are presented in Section 4 and the conclusions are summarized in Section 5.

## 2. Lagrangian decomposition presented in [12]

Mauri and Lorena [12] report several methods to find upper bounds for *QP*. In this section, we describe the one which presents the best results and will be used as base for our column generation approach.

Given a matrix of real numbers $\mathbf{Q}=[q_{ij}]_{m\times m}$, the *QP* can be formulated by the expression

$$QP:\ v(QP)=\max_{\mathbf{x}\{0,1\}}\in\sum_{i=1}^{m}\sum_{j=1}^{m}q_{ij}x_ix_j \tag{1}$$

Applying a simple linearization technique in (1), the quadratic terms $x_ix_j$ can be replaced by the continuous variable $w_{ij}$ (if $i\neq j$) and constraints that guarantee the equality $w_{ij}=x_ix_j$ are also inserted in the model. The quadratic terms $x_ix_j$ can be replaced by $x_i$ if $i=j$. Therefore, we get a mixed integer linear model for *QP* (2)–(7). This model will be denoted by *LQP*.

$$LQP:\ v(LQP)=\max\sum_{i=1}^{m}q_{ii}x_i+\sum_{i\neq j}q_{ij}w_{ij} \tag{2}$$

**Subject to**:

$$w_{ij}-x_i\leq 0,\quad i\neq j,\ q_{ij}>0 \tag{3}$$

$$w_{ij}-x_j\leq 0,\quad i\neq j,\ q_{ij}>0 \tag{4}$$

$$x_i+x_j-w_{ij}\leq 1,\quad i\neq j,\ q_{ij}<0 \tag{5}$$

$$w_{ij}\geq 0,\quad i\neq j,\ q_{ij}<0 \tag{6}$$

$$x_i\in\{0,1\},\quad i=1,\ldots,m \tag{7}$$

A graph $G=(V,E)$ with $V=\{1,\ldots,m\}$ and an adjacency matrix $\mathbf{E}=[e_{ij}]_{m\times m}$, $e_{ij}=1$ if $q_{ij}\neq 0$ and $e_{ij}=0$ if $q_{ij}=0$ can be constructed through the matrix $\mathbf{Q}$. This graph can be partitioned into $n$ ($n\leq m$) independent clusters of vertices with $V=V_1\cup V_2\cup\ldots\cup V_n$, and $V_i\cap V_j=\emptyset$, for $i,j=1,\ldots,n$, $i\neq j$; $G_i=(V_i,E_i)$, $i=1,\ldots,n$, and $X_i=V-V_i, i=1,\ldots,n$. So, the model *LQP* can be rewritten as follows:

$$LQP^n:\ v(LQP^n)=\max\sum_{k=1}^{n}\left(\sum_{i\in V_k}q_{ii}x_i+\sum_{\substack{i\in V_k;\\ j\in V_k;\\ i\neq j}}q_{ij}w_{ij}+\sum_{\substack{i\in V_k;\\ j\in X_k;\\ i\neq j}}q_{ij}w_{ij'}\right) \tag{8}$$

**Subject to**:

$$w_{ij}-x_i\leq 0,\quad i\in V_k,\ j\in V_k,\ i\neq j,\ q_{ij}>0,\quad k=1,\ldots,n \tag{9}$$

$$w_{ij}-x_j\leq 0,\quad i\in V_k,\ j\in V_k,\ i\neq j,\ q_{ij}>0,\ k=1,\ldots,n \tag{10}$$

$$x_i+x_j-w_{ij}\leq 1,\quad i\in V_k,\ j\in V_k,\ i\neq j,\ q_{ij}<0,\ k=1,\ldots,n \tag{11}$$

$$w_{ij}\geq 0,\quad i\in V_k,\ j\in V_k,\ i\neq j,\ q_{ij}<0,\ k=1,\ldots,n \tag{12}$$

$$w_{ij'}-x_i\leq 0,\quad i\in V_k,\ j\in X_k,\ i\neq j,\ q_{ij}>0,\ k=1,\ldots,n \tag{13}$$

$$w_{ij'}-x_{j'}^k\leq 0,\quad i\in V_k,\ j\in X_k,\ i\neq j,\ q_{ij}>0,\ k=1,\ldots,n \tag{14}$$

$$x_i+x_{j'}^k-w_{ij'}\leq 1,\quad i\in V_k,\ j\in X_k,\ i\neq j,\ q_{ij}<0,\ k=1,\ldots,n \tag{15}$$

$$w_{ij'}\geq 0,\quad i\in V_k,\ j\in X_k,\ i\neq j,\ q_{ij}<0,\ k=1,\ldots,n \tag{16}$$

$$x_j-x_{j'}^k=0,\quad j\in X_k,\ k=1,\ldots,n \tag{17}$$

$$w_{ij'}-w_{ji'}=0,\quad i\in V_k,\ j\in X_k,\ j>i,\ k=1,\ldots,n \tag{18}$$

$$x_i\in\{0,1\},\quad i\in V_k,\ k=1,\ldots,n \tag{19}$$

The variables $x_{j'}^k$ represent the copies of the vertex $j$ ($j'$) in the cluster $k$, and the variables $w_{ij'}$ represent the edges between the vertices $i$ and $j'$ (copy of $j$). The constraints (9)–(12) are related to the edges $(i,j)$ with both vertices inside the cluster (sub-graph) $k$, and the constraints (13)–(16) are related to edges $(i,j)$ with vertices in different clusters (inter-clusters edges). The constraints (17) and (18) are copy constraints that ensure the equality between the original and copies variables.

Constraints (17) and (18) can be relaxated in the Lagrangian way with multiplier column vectors $\vec{\alpha}$ and $\vec{\beta}$ ($\vec{\alpha}$ and $\vec{\beta}$ unrestricted). So, the problem $LQP^n$ (indirectly *QP*) can be partitioned into $n$ independent subproblems, and for each subproblem $k$ ($k=1,\ldots,n$) we have the following model:

$$LD_{\alpha\beta}LQP_k:\quad v(LD_{\alpha\beta}LQP_k)=\max\sum_{i\in V_k}\left(q_{ii}-\sum_{d\neq k}\alpha_i^d\right)x_i+\sum_{j\in X_k}\alpha_j^kx_j^k$$
$$+\sum_{\substack{i\in V_k\\ j\in V_{k'}\\ j\neq i}}q_{ij}w_{ij}+\sum_{\substack{i\in V_k\\ j\in X_{k'}\\ j>i}}(q_{ij}-\beta_{ij})w_{ij'}+\sum_{\substack{i\in V_k\\ j\in X_{k'}\\ j<i}}(q_{ij}+\beta_{ji})w_{ij'} \tag{20}$$

**Subject to** : (9),(10),...,(15) and (19) (removing, $k=1,\ldots,n$)

$$w_{ij'}\in\{0,1\},\quad i\in V_k,\quad j\in X_k,\quad i\neq j,\ q_{ij}<0 \tag{21}$$

$$x_j^k\in\{0,1\},\quad j\in X_k \tag{22}$$

Constraints (21) and (22) are inserted into the model to ensure the feasibility of subproblems. Now, the *LQP* relaxation in $n$ subproblems is given by expression (23) and the corresponding Lagrangian dual is presented in expression (24):

$$LD_{\alpha\beta}LQP^n:\ v(LD_{\alpha\beta}LQP^n)=\sum_{k=1}^{n}v(LD_{\alpha\beta}LQP_k) \tag{23}$$

$$DLD_{\alpha\beta}LQP^n:\ v(DLD_{\alpha\beta}LQP^n)=\min_{\vec{\alpha},\vec{\beta}\ \text{unrestricted}}\{v(LD_{\alpha\beta}LQP^n)\} \tag{24}$$

Mauri and Lorena [12] used a subgradient algorithm to solve the Lagrangian dual (24), and CPLEX was used to solve each subproblem $k$ (20)–(22) for finding the decomposition value (23) at the iterations in the subgradient algorithm. In addition, the graph $G$ was partitioned by METIS [13].

## 3. Proposed column generation

Using the Lagrangian decomposition model presented in the previous section, we propose a column generation approach as an alternative to find the Lagrangian multipliers for solving the dual problem (24). To facilitate the understanding of the proposed method, the model (8)–(19) can be rewritten using a matrix notation as follows:

$$LQP^n:\ v(LQP^n)=\max\sum_{k=1}^{n}(\mathbf{q}_{\mathbf{x}}^k\mathbf{x}^k+\mathbf{0}\mathbf{x}'^k+\mathbf{q}_w^k\mathbf{w}^k+\mathbf{q}_{w'}^k\mathbf{w}'^k) \tag{25}$$

$$\textbf{Subject to}: \quad \begin{bmatrix} \mathbf{A_1} & \mathbf{A_2} & \cdots & \mathbf{A}_n \\ \mathbf{B_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B_2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_n \end{bmatrix} \times \begin{bmatrix} \begin{bmatrix} \mathbf{x}^1 \\ \mathbf{x}'^1 \\ \mathbf{w}^1 \\ \mathbf{w}'^1 \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \mathbf{x}^n \\ \mathbf{x}'^n \\ \mathbf{w}^n \\ \mathbf{w}'^n \end{bmatrix} \end{bmatrix} \approx \begin{bmatrix} \mathbf{b}^{\mathrm{A}} \\ \mathbf{b}^{\mathrm{B}} \end{bmatrix} \tag{26}$$

where

- $\mathbf{q}_x^k$: row vector with coefficients of the variables $x_i$, $i \in V_k$;
- $\mathbf{q}_w^k$: row vector with coefficients of the variables $w_{ij}$, $i \in V_k$ and $j \in V_k$;
- $\mathbf{q}_{w'}^k$: row vector with coefficients of the variables $w_{ij'}$, $i \in V_k$ and $j \in X_k$;
- $\mathbf{B}_k$: matrix with coefficients of the variables on constraints (9)–(16);
- $\mathbf{A}_k$: matrix with coefficients of the variables on copy constraints (17) and (18);
- $\mathbf{x}^k$: column vector with decision variables $x_i$, $i \in V_k$;
- $\mathbf{x}'^k$: column vector with decision variables $x_j^k$, $j \in X_k$;
- $\mathbf{w}^k$: column vector with decision variables $w_{ij}$, $i \in V_k$ and $j \in V_k$;
- $\mathbf{w}'^k$: column vector with decision variables $w_{ij'}$, $i \in V_k$ and $j \in X_k$;
- $\approx$: relational operators ($\leq$, $\geq$ or $=$) for each constraint (9)–(18);
- $\mathbf{b}^{\mathrm{A}}$: column vector with right side values (0) for constraints (17) and (18).
- $\mathbf{b}^{\mathrm{B}}$: column vector with right side values (0 or 1) for constraints (9)–(16).

Considering $R_k$ as the constraints set for the subproblem $k$ ($k = 1, \ldots, n$), and $\mathbf{x}^k$, $\mathbf{x}'^k$, $\mathbf{w}^k$ and $\mathbf{w}'^k$ satisfying $R_k$, we can rewrite the model (20)–(22) as follows:

$$LD_{\alpha\beta}LQP_k: \quad v(LD_{\alpha\beta}LQP_k) = \mathbf{max}\left\{ \left( [\mathbf{q}_x^k \ \mathbf{0} \ \mathbf{q}_w^k \ \mathbf{q}_{w'}^k] - \begin{bmatrix} \vec{\alpha} \\ \vec{\beta} \end{bmatrix}^{\mathrm{T}} \right) \begin{bmatrix} \mathbf{x}^k \\ \mathbf{x}'^k \\ \mathbf{w}^k \\ \mathbf{w}'^k \end{bmatrix} \right\} \tag{27}$$

The classical column generation technique uses a coordinator problem – or restricted master problem (*RMP*) – and subproblems for column generation to *RMP*. The *RMP* uses dual variables for guiding subproblems on the search for new columns. So, applying the *Dantzig–Wolfe* decomposition on the model $LQP^n$ we have

$$RMP: \quad v(RMP) = \mathbf{max} \sum_{k=1}^{n} \sum_{s \in S_k} \lambda_{sk} (\mathbf{q}_x^k \mathbf{x}^{sk} + \mathbf{0}\mathbf{x}'^{sk} + \mathbf{q}_w^k \mathbf{w}^{sk} + \mathbf{q}_{w'}^k \mathbf{w}'^{sk}) \tag{28}$$

**Subject to**:

$$\sum_{k=1}^{n} \sum_{s \in S_k} \lambda_{sk} \left( \mathbf{A}_k \begin{bmatrix} \mathbf{x}^{sk} \\ \mathbf{x}'^{sk} \\ \mathbf{w}^{sk} \\ \mathbf{w}'^{sk} \end{bmatrix} \right) \approx \mathbf{b}^{\mathrm{A}} \tag{29}$$

$$\sum_{s \in S_k} \lambda_{sk} = 1 \quad \forall k \in \{1, \ldots, n\} \tag{30}$$

$$\lambda_{sk} \geq 0 \quad \forall k \in \{1, \ldots, n\},\ s \in S_k \tag{31}$$

The extreme points of a polyhedral for the cluster (subproblem) $k$ is given by the set $S_k$. $\mathbf{x}^{sk}$, $\mathbf{x}'^{sk}$, $\mathbf{w}^{sk}$ and $\mathbf{w}'^{sk}$ are vectors with the decision variables values from the $s$-th column inserted into *RMP*, i.e., $s$-th extreme point of a polyhedral for the cluster $k$ (feasible solution for the subproblem $k$). $\lambda_{sk}$ is the decision variable for the $s$-th extreme point in $S_k$. For each subproblem $k$ (27), the Lagrangian multiplier vectors $\vec{\alpha}$ and $\vec{\beta}$ can be considered (replaced) as the dual variables vectors related to constraint (29). For notation, we will still treat these vectors as $\vec{\alpha}$ and $\vec{\beta}$, but now considering them as dual variables vectors.

Considering that $\mu_k$ is the dual variable for $k$-th convexity constraint of (30), a new column for the subproblem $k$ is inserted into *RMP* if the reduced cost $\theta_k$ is positive ($\theta_k = v(LD_{\alpha\beta}LQP_k) - \mu_k > 0$). So, the *RMP* guides the subproblem solutions through the dual variables looking for a good solution for the original problem.

The initial *RMP* is constructed by the heuristic presented in Fig. 1. This heuristic uses the local search introduced in [14]. Fig. 2 presents the proposed algorithm: new columns are generated until satisfying a stop criteria and the final *RMP* formed by all of the generated columns is solved with integers ($\lambda_{sk} \in \{0, 1\}$). Integer solutions will be feasible for *LQP* and indirectly for *QP*. The *RMP* in integers will be denoted $RMP_{int}$, and the value of its solution, $v(RMP_{int})$, will be a lower bound (approximate solution) for *QP*.

The linear relaxation of *RMP*, $v(RMP)$, can be a valid upper bound to *QP* at the end of the column generation algorithm, after inserting sufficient columns, but the solution of the Lagrangian dual, $v(DLD_{\alpha\beta}LQP^n)$, will be always an upper bound for *QP*. Our column generation algorithm stops when an optimal solution for the relaxed model ($LQP^n$) is found. So, we can verify the quality of the upper bound from our Lagrangian decomposition. In addition, we have also an approximate solution for the original model *QP*.

```
1  for v ← 1 to m do
2  |  x_v ← 1;
3  |  Apply a local search to find a good solution with x_v = 1;
4  |  for each variable w_ij do
5  |  |  if (x_i = 1) and (x_j = 1) then
6  |  |  |  w_ij ← 1;
7  |  |  else
8  |  |  |  w_ij ← 0;
9  |  |  end
10 |  end
11 |  Distribute the variables of the found solution in the clusters;
12 |  Set the copies variables with the values from the original ones;
13 |  Create a column for each cluster with the values from the solution;
14 |  Insert the new columns in RMP;
15 end
```

**Fig. 1.** Initial *RMP*.

```
1  Apply the METIS heuristic to split the graph and define the clusters;
2  Decompose the original problem and mount the subproblems;
3  Find a set of initial columns to form a RMP (Figure 1);
4  Optimize the RMP (by CPLEX) to get a set of optimal dual variables;
5  Update and solve the subproblems (27) through CPLEX;
6  if (∀k, θ_k ≤ 0) or v(DLD_αβLQP^n) = v(RMP) then
7  |  Stop: an optimal solution for RMP was found;
8  |  Solve RMP_int through CPLEX to find v(RMP_int);
9  else
10 |  Insert all of columns with θ_k > 0 in RMP;
11 |  Back to step 4;
12 end
```

**Fig. 2.** Proposed *CG*.

## 4. Computational results

Computational experiments were performed over a set of 45 hard instances available at *OR-Library* (http://people.brunel.ac.uk/~mastjjb/jeb/info.html). These instances were formed by Pardalos and Rodgers [2] generator and they had been separated into six classes (A, B, C, D, E and F) with different features ($m$, density, and values'

range for matrix **Q**). The problems of classes A, B and C were introduced in [2], and the others in [15]. According to Glover et al. [15], these instances have been among the most difficult ones found in the literature.

As reported in [12], we split the graph $G$ by METIS heuristic [13], and we use CPLEX to solve the *RMP* and the subproblems (both within a limit time of 1 h for each instance). The number of clusters ($n$) for each instance are similar to the ones presented in [12].

Table 1 reports the average gaps for each class of instances. These gaps are presented for the proposed *CG* and other relaxations found in the literature, and they were computed as shown in expression (32). $v(OPT)$ represents the best known solutions reported in [15], and $v(Bound)$ is the value for the upper bounds obtained by the cited methods.

$$Gap = \frac{v(Bound) - v(OPT)}{v(OPT)} \times 100 \quad (32)$$

In Table 1, the line $m$ reports the instances' size variation, and the line *Dsty* presents the range of the density for matrix **Q**. Next lines are related to [12], and we have the following: $\overline{LQP}$ indicates the *roof dual* (linear relaxation of *LQP*); *LQPC* is the *LQP* linear relaxation using cut constraints; *Best lag* shows the best gaps from traditional Lagrangian relaxations (relaxing constraints (3), (4) and/or (5) without clusters partitioning); and the next four lines report the gaps from different Lagrangian decomposition approaches. The last lines report the gaps from our *CG* approach, i.e., the dual of the Lagrangian decomposition ($DLD_{\alpha\beta}LQP^n$) and *RMP*. The last columns present the average times for getting the upper bounds to each class of instance. The best gaps are highlighted in bold.

Table 2 reports the gaps from the integer solutions for the instances of classes B, D and F. The gaps obtained from the proposed *CG* $(((v(OPT) - v(RMP_{int}))/v(OPT)) * 100)$ are compared against the ones presented by the different approaches described in [6,1] and by CPLEX. The methods reported in [12] were not inserted in Table 2 because they present only upper bounds for *QP*, i.e., they do not search for integer solutions.

Our proposed *CG* uses average times of 96.42, 4522.32 and 4455.33 s for the instances of classes B, D and F respectively.

**Table 1**
Average gaps (%) from the upper bounds for all class of instances.

| **Instances class** | | **A** | **B** | **C** | **D** | **E** | **F** | **Average time (s)** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ***m*** | | 30–100 | 20–125 | 40–100 | 100 | 200 | 500 | | | | | | |
| **Dsty (%)** | | 6.25–50 | 10–100 | 10–80 | 10–100 | 10–50 | 10–100 | **A** | **B** | **C** | **D** | **E** | **F** |
| **Mauri and Lorena** [12] | $\overline{LQP}$ | 9.99 | 656.34 | 33.31 | 155.37 | 160.56 | 159.28 | 0.01 | 0.06 | 0.04 | 0.80 | 2.72 | 1565.53 |
| | *LQPC* | 6.16 | 404.22 | 15.11 | 85.35 | 95.43 | 126.12 | 0.02 | 4.92 | 0.63 | 68.10 | 311.07 | 2889.70 |
| | *Best lag* | 2.98 | 657.66 | 9.71 | 111.40 | 136.72 | 124.66 | 19.84 | 2.14 | 2981.73 | 3799.08 | 5135.07 | 4510.44 |
| | *Dec*1 | 0.04 | 29.85 | **0.00** | 98.78 | 92.91 | 95.68 | 7.76 | 480.38 | 135.15 | 5446.11 | 3975.54 | 3662.57 |
| | *Dec*2 | 0.06 | 29.86 | **0.00** | 21.54 | 69.33 | 93.44 | 12.40 | 349.89 | 14.85 | 4981.19 | 3651.19 | 3615.65 |
| | *Dec*3 | **0.00** | 10.33 | **0.00** | 4.76 | 78.52 | 92.52 | 4.40 | 700.65 | 78.04 | 6356.60 | 3958.04 | 3909.34 |
| | *Dec*3* | 0.04 | 15.43 | **0.00** | 8.32 | 81.81 | 98.46 | 16.03 | 822.87 | 85.10 | 6131.73 | 4262.61 | 4088.06 |
| **Proposed *CG*** | $DLD_{\alpha\beta}LQP^n$ | 2.22 | 323.72 | 3.41 | 109.87 | 713.29 | 1102.40 | | | | | | |
| | *RMP* | 0.17 | **3.44** | 0.03 | **0.11** | **0.05** | **0.38** | 6.95 | 96.41 | 226.18 | 4522.32 | 62.76 | 4455.33 |

**Table 2**
Gaps (%) from the integer solutions for the instances of classes B, D and F.

| **Inst.** | *m* | **Dsty** (%) | **Best known** | **Glover et al.** [6] | | | | | **Billionnet and Elloumi** [1] | **CPLEX** | **Proposed *CG*** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | *DDT* | *A2n* | *A2t* | *V3n* | *V3t* | | | |
| 1b | 20 | 100 | 133 | 26.30 | **0.00** | 21.10 | 89.50 | 89.50 | **0.00** | **0.00** | **0.00** |
| 2b | 30 | 100 | 121 | 5.00 | 24.80 | 13.20 | 95.00 | 95.00 | **0.00** | **0.00** | **0.00** |
| 3b | 40 | 100 | 118 | 52.50 | 13.60 | 19.50 | 100.00 | 100.00 | **0.00** | **0.00** | **0.00** |
| 4b | 50 | 100 | 129 | 33.30 | 21.70 | 21.70 | 100.00 | 100.00 | **0.00** | **0.00** | **0.00** |
| 5b | 60 | 100 | 150 | 30.00 | **0.00** | 40.00 | 99.30 | 99.30 | **0.00** | **0.00** | **0.00** |
| 6b | 70 | 100 | 146 | 56.80 | 22.60 | 27.40 | 97.30 | 97.30 | **0.00** | **0.00** | 12.30 |
| 7b | 80 | 100 | 160 | 43.70 | **0.00** | **0.00** | 99.40 | 99.40 | **0.00** | **0.00** | **0.00** |
| 8b | 90 | 100 | 145 | 38.60 | 19.30 | 19.30 | 100.00 | 100.00 | **0.00** | **0.00** | **0.00** |
| 9b | 100 | 100 | 137 | 29.90 | 7.30 | 24.10 | 100.00 | 100.00 | **0.00** | **0.00** | 2.90 |
| 10b | 125 | 100 | 154 | 34.40 | 21.40 | 21.40 | 100.00 | 100.00 | **0.00** | **0.00** | 4.50 |
| 1d | 100 | 10 | 6333 | – | – | – | – | – | 4.10 | 0.00 | **0.00** |
| 2d | 100 | 20 | 6579 | – | – | – | – | – | 10.00 | 5.21 | **0.00** |
| 3d | 100 | 30 | 9261 | – | – | – | – | – | 7.60 | 3.54 | **0.00** |
| 4d | 100 | 40 | 10 727 | – | – | – | – | – | 8.10 | 6.25 | **0.00** |
| 5d | 100 | 50 | 11 626 | – | – | – | – | – | 8.70 | 17.82 | **0.19** |
| 6d | 100 | 60 | 14 207 | – | – | – | – | – | 7.20 | 9.90 | **0.35** |
| 7d | 100 | 70 | 14 476 | – | – | – | – | – | 8.30 | 14.02 | **0.00** |
| 8d | 100 | 80 | 16 352 | – | – | – | – | – | 6.10 | 17.48 | **0.00** |
| 9d | 100 | 90 | 15 656 | – | – | – | – | – | 8.70 | 14.65 | **0.06** |
| 10d | 100 | 100 | 19 102 | – | – | – | – | – | 6.90 | 11.99 | **0.00** |
| 1f | 500 | 10 | 61 194 | 0.80 | 22.10 | 23.10 | 7.10 | 7.20 | – | 42.50 | **0.38** |
| 2f | 500 | 25 | 100 161 | 0.70 | 19.60 | 19.20 | 7.30 | 9.60 | – | 84.00 | – |
| 3f | 500 | 50 | 138 035 | 1.30 | 21.40 | 20.30 | 7.30 | 7.90 | – | 76.40 | – |
| 4f | 500 | 75 | 172 771 | 1.30 | 18.00 | 16.10 | 5.60 | 5.60 | – | 84.50 | – |
| 5f | 500 | 100 | 190 507 | 1.10 | 12.20 | 22.50 | 4.60 | 5.80 | – | 89.50 | – |

The method presented in [1] was not able to solve the instances of class F, and it uses an average time of 600 s for the instances of class D. In addition, the authors indicate that all optimal solutions were found for the instances of class B within an average time of 7200 s. CPLEX runs for an average of 3600 s for the instances of classes D and F, and 3 s for the class B.

For the instances of classes A, C and E, we can say that: the method proposed in [1] found the optimal solutions for classes A and C within 7200 s, but it was not able to solve the instances of class E; CPLEX uses an average of 3.28 s for the instances of classes A and C, and it presents an average gap of 176.86% for class E within 3600 s; finally, our proposed *CG* found the optimal solutions for all of the instances of classes A, C and E within average times of 6.95, 226.18 and 62.76 s respectively.

All of our experiments were implemented in C++ and performed in a PC with AMD Athlon of 2.2 GHz processor with 1 GB of RAM memory. That is the same computer used in [12]. Billionnet and Elloumi [1] have used a PC with processor Intel Pentium IV of 1.6 GHz with 1 GB of RAM memory, and Glover et al. [6] do not report the results for the instances of classes A, C, D and E and the computational times for classes B and F. The computer used by Billionnet and Elloumi [1] is different, however, we can see an estimative of the method performance.

## 5. Conclusions

We reported a new strategy to solve a Lagrangian decomposition for the unconstrained binary quadratic programming (*QP*) problem. The proposed method is based on the column generation (*CG*) and gets upper bounds and feasible solutions for *QP*. Hard instances with different features were used to evaluate the proposed method.

Results show that our *CG* is able to improve the convergence time from a Lagrangian decomposition, i.e., we have improved solutions within smaller computational times replacing the subgradient algorithm in the Lagrangian decomposition reported in [12] by our proposed *CG*. In addition, the proposed *CG* was directly compared against other methods reporting good results for practically all of the instances.

Finally, we believe that a more efficient technique to solve the subproblems for column generation can result in better solutions for the instances treated in this paper and possibly for larger ones. Besides, our *CG* can be a good alternative to improve the results for problems with constraints modeled by graphs.

## Acknowledgments

The authors acknowledge the referees for their valuable suggestions that have improved the quality of the paper, and FAPES (process 45391998/09) and CNPq (processes 300747/2010-1, 300692/2009-9 and 470813/2010-5) for the financial support.

## References

[1] Billionnet A, Elloumi S. Using a mixed integer quadratic programming solver for the unconstrained quadratic 0–1 problem. Mathematical Programming 2007;109(1):55–68.
[2] Pardalos PM, Rodgers GP. Computational aspect of a branch and bound algorithm for quadratic 0–1 programming. Computing 1990;45(2):131–44.
[3] Billionnet A, Sutter A. Minimization of a quadratic pseudo-Boolean function. European Journal of Operational Research 1994;78(1):106–15.
[4] Huang HX, Pardalos PM, Prokopyev OA. Lower bound improvement and forcing rule for quadratic binary programming. Computational Optimization and Applications 2006;33(2):187–208.
[5] Rendl F, Rinaldi G, Wiegele A. Solving max-cut to optimality by intersecting semidefinite and polyhedral relaxations. Mathematical Programming 2010;121(2):307–35.
[6] Glover F, Alidaee B, Rego C, Kochenberger G. One-pass heuristics for large-scale unconstrained binary quadratic problems. European Journal of Operational Research 2002;137(2):272–87.
[7] Palubeckis G. Multistart tabu search strategies for the unconstrained binary quadratic optimization problem. Annals of Operations Research 2004;131(1):259–82.
[8] Hansen P, Meyer C. Improved compact linearizations for the unconstrained quadratic 0–1 minimization problem. Discrete Applied Mathematics 2009;157(6):1267–90.
[9] Adams WP, Dearing PM. On the equivalence between roof duality and Lagrangian duality for unconstrained 0–1 quadratic programming problems. Discrete Applied Mathematics 1994;48(1):1–20.
[10] Adams WP, Forrester RJ. Linear forms of nonlinear expressions: new insights on old ideas. Operations Research Letters 2007;35(4):510–8.
[11] Chardaire P, Sutter A. A decomposition method for quadratic zero–one programming. Management Science 1995;41(4):704–12.
[12] Mauri GR, Lorena LAN. Lagrangean decompositions for the unconstrained binary quadratic programming problem. International Transactions in Operational Research 2011;18(2):257–70.
[13] Karypis G, Kumar V. Multilevel *k*-way partitioning scheme for irregular graphs. Journal of Parallel and Distributed Computing 1998;48(1):96–129.
[14] Beasley JE. Heuristic algorithms for the unconstrained binary quadratic programming problem. Technical Report - Management School, Imperial College: London - UK, 1998. Available at: http://people.brunel.ac.uk/~mastjjb/jeb/bqp.pdf.
[15] Glover F, Kochenberger GA, Alidaee B. Adaptative memory tabu search for binary quadratic programs. Management Science 1998;44(3):336–45.